2021-04

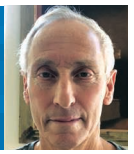
Who Is Authenticating My E-Commerce Logins?

Drusinsky, Doron

IEEE

Drusinsky, Doron. "Who Is Authenticating My E-Commerce Logins?." Computer 54.04 (2021): 49-54.
http://hdl.handle.net/10945/67406

EDITOR **DORON DRUSINSKY**
Naval Postgraduate School ddrusins@nps.edu



Who Is Authenticating My E-Commerce Logins?

Doron Drusinsky, Naval Postgraduate School

Multifactor authentication has inherent security flaws, including its reliance on passwords. A generic solution is proposed for authetication based on a hybrid of biometric-authentication artifacts, digital certificates, and error correction.

Trusting a mobile device using modern public-key cryptography does not assure that the person holding the phone is the same person identified by the certificate being used.

MULTIFACTOR AUTHENTICATION FOR E-COMMERCE USING MOBILE CLIENTS

The best-practice security protocol for mobile device users when authenticating themselves to an online service is multifactor authentication (MFA), which usually is implemented as two-factor authentication (2FA). One example of such 2FA is online banking sign-on using a password, being the *something you know* factor, and a random one-time code (*nonce*) sent to the device via text messaging or email, whereby sending back the nonce the user proves to the bank that the user possesses the *something you have* factor. Another example of 2FA is a similar combination of password and a secure token device that generates tokens locally without transmitting them.

2FA is not ideal for several reasons. First, it involves repeated friction as it is time-consuming and requiring multiple actions on part of the user. 2FA is also a rather weak protocol because passwords are an inherently weak security system[15] and are exposed when used in public or when a malicious keylogger is present. In addition, 2FA is susceptible to the well-known SIM swap attack,[1,13] where attackers use social engineering techniques to convince

Digital Object Identifier 10.1109/MC.2021.3055684
Date of current version: 9 April 2021

the mobile carrier that they are the rightful owner of the cell phone number to obtain a new SIM card. Last but not least, if a mobile device is lost or stolen while open, then the single remaining protection becomes the password component of the 2FA protocol. Secure token devices are also vulnerable because they are not protected and can easily be lost.

Hence, the industry is moving toward cryptographic e-commerce authentication for mobile devices (that is, the Fast Identity Online standard[7]); such authentication protocols rely on digital signatures as follows. During registration, the mobile device obtains a certified key-pair (private and public) from the e-commerce service (for example, a banking service) or some trusted certification authority (CA). To authenticate to an e-commerce session, the service provider sends the device a nonce challenge, the device signs it with its private key, and the service validates the signature using the public key. Such a protocol is both more secure and also seamless to the mobile device owner.

To authenticate to an e-commerce session, the service provider sends the device a nonce challenge, the device signs it with its private key, and the service validates the signature using the public key.

As an example, consider authenticating for your online banking system. With a digital signature-based authentication method, you will start by opening your banking application on your mobile device. The app will validate your presence using biometric authentication and then prove to the banking service that your device holds the (secret) private key associated with the public key registered with your account. Proof is accomplished using the challenge-response protocol described previously.

Clearly, since a key-pair is just a pair of mathematically related numbers, a signature on its own does not imply that any mobile device in particular signed the given document: hence enter digital certificates. A certificate is signed by a trusted CA (trusted by the verifier); it is the manifestation of trust, effectively saying that "the trusted authority verified that person/company with identity *x* is associated with public key *y*." A digital certificate ties such a trusted identity of the individual that owns the device to the public key listed in the certificate. Since that public key is associated with a unique private key, then the certificate effectively binds all three artifacts together, namely a private key (resides on device), a public key (easily computable from the private key, also resides in the certificate,) and the identity of the certificate holder.

However, as mentioned earlier, people do not sign digital documents, devices do. Hence, for example, if a device is lost or stolen, then absent further protection, the thief can authenticate on behalf of the owner. For this reason, digital signatures performed on mobile devices are typically temporally preceded by biometric authentication. Nevertheless, biometric authentication can be reduced to password authentication (after multiple trial and errors), thereby reducing the entire e-commerce authentication process to one-factor authentication (1FA), which is the device's password. Moreover, the e-commerce service provider cannot guarantee that the application performing biometric authentication on its behalf is the intended authenticator because any mobile application can sign using private keys on the same device.

A note about password security, whether for authenticating into the mobile device or as part of e-commerce MFA, is due. Most e-commerce services have built-in procedures that promote safe password usage, such as periodic password changes and password complexity tests. The flipside of strong and fresh password maintenance, however, is increased friction, where the user needs to remember and enter many different passwords on his/her mobile devices. As a result, some scary statistics about password reuse have emerged, such as that 65% of users reuse passwords or that an average person reuses as password on as many as 14 times. In short, although e-commerce services try to enforce password security, end users are not complying that well.

The underlying problem is, therefore, how can one better ensure that the person whose trusted identity appears in the certificate is the person holding the device while the device is signing a nonce or any other document?

This article proposes a generic solution based on a hybrid of biometric authentication artifacts, digital certificates, and error correction.

BIOMETRIC AUTHENTICATION FOR MOBILE DEVICES

Biometric authentication relies on the unique biological characteristics of the individual holding the device to verify that he or she is the person that originally trained the device. Biometric authentication systems use varying levels of machine learning (ML) to compare runtime biometric data (henceforth, biometrics) to biometrics collected during training.

Numerous commercially available mobile device biometric systems exist on the market, including the following:

- fingerprint scanning,[6] which authenticates using patterns of raised areas and branches in the individual's finger image
- finger vein ID,[14] which authenticates using vascular pattern in the individual's finger

- facial recognition, which authenticates using numeric codes for nodal points on a human face called faceprints
- voice verification, which authenticates using characteristics of the individual's voice
- retina scans,[5] which authenticates using blood vessel patterns in the light-sensitive surface lining the individual's inner eye
- iris recognition, which authenticates using unique patterns within the ring-shaped region surrounding the pupil of the eye
- neuro print,[9,10] which authenticates using micro-vibrational patterns in the user's hands.

Often, individual biometric data differ from one reading to another. Consider voice verification. For example, if an individual says "I am John, John Doe" on two different recordings, there are bound to be differences between the two, such as speed, pitch, and sometimes even more profound changes due to allergies or weather sensitivities. Hence, many of these biometric authentication systems rely on signal processing and subsequent ML to authenticate.

Mobile ML systems for biometric authentication are initially trained by the device owner before being used for runtime authentication. Both phases rely on features (numeric values such as faceprints) extracted from sensors on the device, while the owner is holding the device. The training phase uses features to train one or more ML algorithm such as neural networks or random decision forests (RDFs),[11] whereas the classification phase inputs features to trained ML models to decide whether the individual holding the phone is the same individual that trained the system.

Each authenticator produces a Boolean output. Some ML techniques, such as RDFs, use batteries of relatively small authenticators (for example, random decision trees) called *bit-classifiers*, thereby yielding a vector of bits *Vt*, as illustrated in Figure 1. The overall authenticator (that is, the RDF) accepts or rejects the input using a simple metric such as a majority vote.

The ML process illustrated in Figure 1 operates as follows. In the training phase, the ML system trains the individual bit-classifiers using a relatively large training set. The classifiers are trained so that their majority vote for end user acceptance and their majority vote for prospective attacker rejection achieve desired levels according to various metrics such as precision, recall, or F1 score.[8] In the runtime phase, a single vector of features is presented to the battery of classifiers, and the single majority vote determines the outcome of the authentication.

In our proposed technique detailed below, we use such batteries of bit-classifiers, but without the majority vote step.

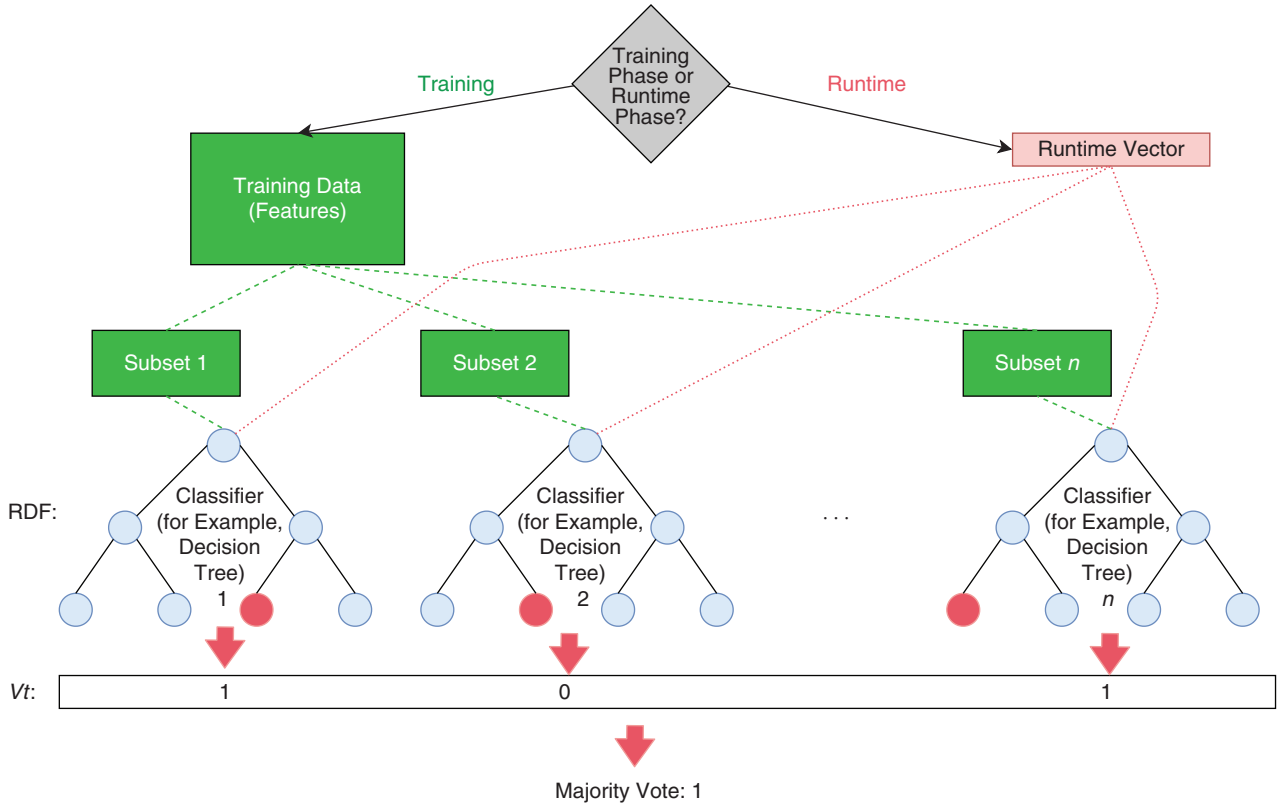


FIGURE 1. The ML process using batteries of bit-classifiers.

ERROR CORRECTION CODES

Error correction is primarily used in telecommunication applications, where a sender encodes the message with redundant information in the form of an error correction code (ECC). The redundancy allows the receiver to correct a limited number of errors that may occur anywhere in the message.

The two main categories of ECCs are block codes and convolutional codes. Block codes work on fixed-size blocks of bits of predetermined size. Practical block codes can generally be hard decoded in polynomial time to their block length. Convolutional codes work on bit or symbol streams of arbitrary length. The block size of a convolutional code is generally arbitrary, while block codes have a fixed size dictated by their algebraic characteristics.

With the proposed application, the vector Vt of Figure 1, obtained during ML training, takes the role of the transmitted message. A corresponding vector Vt^-, obtained during runtime authentication, takes the role of the received message. Error correction is then applied to Vt^- to obtain the original vector Vt.

THE PROPOSED GENERIC SOLUTION

The proposed solution takes part in two phases: 1) during certificate generation and 2) during runtime mobile device authentication.

The proposed certificate generation process

- The device owner trains his or her biometric authentication ML system and generates a battery M-bit-classifier. As illustrated in Figure 1, given a vector of features, this battery induces a corresponding M-bit vector Vt (the size of M is discussed in the "Research Challenges" section).
- The device owner obtains ECC bits for Vt, denoted as $E(Vt)$, using an error correction algorithm of choice such as a Golay block code.[3,4]
- The device owner generates two key-pairs: i) $KP_i = (d_i, Q_i)$, a key-pair associated with owners' identity, where the private key is a nonce, and ii) $KP_b = (d_b, Q_b)$, a key-pair associated with owner's biometric features. (With a contemporary elliptic curve digital signature algorithm (ECDSA), the private key d is a long number (256 bits long), and its corresponding public key Q is a point on the agreed-upon curve.) The private key is $d_b = Vt + r_b$, with Vt being Vt of Figure 1 and r_b being a nonce. The private key d_b, based on Vt of the training set, is called the *golden private key*. Note that two keys are stored in the mobile device's secure key storage: d_i and r_b; Vt is not stored anywhere, that is, it is abandoned.
- The device owner sends a certificate signing request (CSR) to a CA that is trusted by the service provider. The CSR contains the owner's identity, $E(Vt)$—the abovementioned ECC bits and the two public keys Q_i and Q_b.
- The CA verifies the identity and returns a signed certificate.

The proposed runtime authentication process

- The mobile device sends a user id to the service (that is, the online banking service).
- The online service returns two random nonces: R_i and R_b.
- The mobile device signs R_i using d_i.
- The mobile device collects end user features and generates a vector Vt^- of M bit-classifications.
- The mobile device uses $E(Vt)$ taken from his or her certificate and applies error correction to Vt^-, thereby retrieving a corrected *vector* Vt^+ of M bit-classifications.
- The mobile device signs R_b using $d_b^+ = r_b + Vt^+$ as the private key of KP_i. We call this private key the *actual private key* as opposed to the golden private key of the third step in the "Proposed Certificate Generation Process" section.
- The mobile device sends both signed-R_i and signed-R_b, along with the certificate, to the service provider for signature verification.
- The service provider verifies both signatures using the public keys in the end user's certificate.

The reason for creating $d_b^+ = r_b + Vt^+$ rather than using Vt alone is to preclude adversaries from using potential adversarial databases of Vts built with features taken from social platforms (in case of facial recognition systems) or hacked databases (for example, hacked fingerprint databases).

Note that the two kinds of key-pairs (KP_i and KP_b) have very different responsibilities. KP_b is responsible for assuring that the *person* holding the phone is the same person that registered the identity. As for KP_i, there is one such pair per e-commerce service to which the device owner subscribes, such as one for the bank, one for an online retailer, and so on. However, if and when the false negative rate associated with the verification of signed-Rb is noticeable, some applications might choose to turn off that verification requirement when the mobile device is in a safe area, such as at home or at work.

RESEARCH CHALLENGES

- Individual trees in an RDF are rather simple data structures; they cannot be relied on as a good obfuscator. Hence, privacy preserving bit-classifier batteries are preferred.[2]
- When used for authentication, individual bit-classifiers such as RDF trees, are trained to produce ones for the owner

and zeros for the attacker, with a majority vote deciding on the final outcome. In our case, however, these bits should have a more balanced distribution of zeros and ones. Naively, one could simply flip the logic of a one-producing classifier to convert it to a zero-producing one. An attacker, however, can easily distinguish between these bit-classifiers, thereby gaining insight into the underlying code. The research challenge is therefore to build a battery of bit-classifiers where some of the bit-classifiers generate a zero while some generate a one, ensuring that the two types are hard to distinguish without having the owner's biometric data.

- Private keys for a contemporary ECDSA[12] are 256 bits long. Consider, for example, that the Golay[3,4] block error correction code applied to Vt^- in the fifth step of the "Proposed Runtime Authentication Process" section is capable of correcting three errors of 12 data bits (using 11 ECC bits). One can then assume therefore that three of the 12 bits are visible, that is, that entropy exists only in the remaining 9 bits. Therefore, one needs at least 28 error correction blocks, inducing $M = 336$ (28*12) or more bit-classifiers overall. For the sake of a back-of-envelope calculation, assume that the true positive rate (TPR) of all of the M bit-classifiers is $p = 0.9$ and that all bit-classifiers are independent. The probability of four to 12 errors in one ECC will then be $\sum_{k=4}^{12} = \binom{12}{k}(1-p)^k p^{12-k} = 0.026$, and therefore the probability of all 28 ECCs being able to correct properly is $(1 - 0.026)^{28} = 0.48$, which is rather low. With a TPR of 0.95, however, that last number goes up to 0.94, which will induce some friction (multiple authentication attempts) but not overwhelming friction.

On the flip side, a false positive rate of f = 0.1 will induce a miniscule probability (1.41*^{10} – 190) of an adversary using his or her own features to create a vector Vt that has three or fewer errors in all 28 blocks.

In summary, to generate a reliable private key d_b using ECDSA, a large collection of high-quality bit-classifiers is required. While some biometric systems might be capable of producing more high-quality bit-classifiers than others, it is highly nontrivial to create or more bit-classifiers with such high TPRs.

- Absent a comprehensive scientific comparison of the entropy content of various biometric classification techniques, one can only make the general observation that there are likely entropy differences between the various biometric techniques. Some techniques such as facial recognition are more susceptible to attacks that use publicly available user data. Other techniques such as neuro print have no available databases, public or otherwise; moreover, neuro-print[2,3] exhibits far more chaotic signal behavior than facial features for example.
- As described in the first step in the section "The Proposed Certificate Generation Process," the vector Vt is extracted from the training set. However, *every* vector of features induces its own vector Vt. It is unclear which such vector, or combination thereof, is the one that makes the overall system work the best.
- Adversarial ML techniques exploit the way that ML algorithms work to disrupt the behavior of artificial intelligence algorithms; the adversarial robustness of ML algorithms is an active area of research. An obvious research challenge is, therefore, the identification of adversarial attacks on the proposed battery of biometric classifiers.

ACKNOWLEDGMENT

The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright annotations thereon.

REFERENCES

1. P. D. Babu, C. Pavani, and C. E. Naidu, "Cyber Security with IOT," in *Proc. 2019 Fifth Int. Conf. Sci. Technol. Eng. Math. (ICONSTEM)*, Chennai, India, pp. 109–113. doi: 10.1109/ICONSTEM.2019.8918782.
2. S. Fletcher and M. Z. Islam, "Differentially private random decision forests using smooth sensitivity," *Expert Syst. Appl.*, vol. 78, pp. 16–31, July 2017. doi: 10.1016/j.eswa.2017.01.034.
3. M. J. E. Golay, "Complementary series," *IRE Trans. Inf. Theory*, vol. 7, no. 2, pp. 82–87, 1961. doi: 10.1109/TIT.1961.1057620.
4. R. J. Higgs and J. F. Humphreys, "Decoding the ternary Golay code," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 1043–1046, May 1993. doi: 10.1109/18.256511.
5. N. K. Shaydyuk and T. Cleland, "Biometric identification via retina scanning with liveness detection using speckle contrast imaging," in *Proc. 2016 IEEE Int. Carnahan Conf. Secur. Tech. (ICCST)*, Orlando, FL, 2016, pp. 1–5. doi: 10.1109/CCST.2016.7815706.
6. D. Valdes-Ramirez et al., "A review of fingerprint feature representations and their applications for latent fingerprint identification: Trends and evaluation," *IEEE Access*, vol. 7, pp. 48,484–48,499, 2019. doi: 10.1109/ACCESS.2019.2909497.

7. Fido Alliance. https://fidoalliance.org/ (accessed Mar. 2021).
8. "Classification: Precision and recall." Google. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall (accessed Mar. 2021).
9. M. Zizi, N. Sullivan, L. Stork, D. Drusinsky, and K. Lee, "Mobile E-commerce authentication using digital signatures," U.S. Patent 63130406, Dec. 23, 2020.
10. Local user authentication with neuro and neuro-mechanical fingerprints, by M. Zizi and H. Sharkey. (2017). U.S. Patent 10 061 911 B2 [Online]. Available: https://patents.google.com/patent/US10061911B2/en?oq=US10061911
11. Classification technique using random decision forests, by T. K. Ho. (1999) U.S. Patent 6 009 199 A [Online]. Available: https://patents.google.com/patent/US6009199A/en
12. "Elliptic curve digital signature algorithm." Wikipedia. https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm (accessed Mar. 2021).
13. "SIM swap scam." Wikipedia. https://en.wikipedia.org/wiki/SIM_swap_scam
14. "Finger vein authentication technology." Hitachi. http://www.hitachi.co.jp/products/it/veinid/global/introduction/fingervein.html (accessed Mar. 2021).
15. Verizon. "2017 data breach investigations report." Knowbe4. https://www.knowbe4.com/hubfs/rp_DBIR_2017_Report_execsummary_en_xg.pdf (accessed Mar. 2021).

DORON DRUSINSKY is a professor in the Naval Postgraduate School's Department of Computer Science, Monterey, California, 93943, USA, and the chief science officer at Aerendir, Inc., Mountain View, California, 94040, USA. Contact him at ddrusins@nps.edu.

Digital Object Identifier 10.1109/MC.2021.3066063